IN-37-CR
126010
46P

JPL Publication 88-6

Recursive Forward Dynamics for Multiple Robot Arms Moving a Common Task Object

G. Rodriguez

(NASA-CR-182526) RECURSIVE FORWARD DYNAMICS FOR MULTIPLE ROBOT ARMS MOVING A COMMON TASK OBJECT (Jet Propulsion Lab.) 40 p CSCL 13I

N88-18016

Unclas
G3/37 0126010

February 15, 1988

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

JPL Publication 88-6

Recursive Forward Dynamics for Multiple Robot Arms Moving a Common Task Object

G. Rodriguez

February 15, 1988

NASA
National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

ABSTRACT

Recursive forward dynamics algorithms are developed for an arbitrary number of robot arms moving a commonly held object. The multiarm forward dynamics problem is to find the angular accelerations at the joints and the contact forces that the arms impart to the task object. The problem also involves finding the acceleration of this object. The multiarm forward dynamics solutions provide a thorough physical and mathematical understanding of the way several arms behave in response to a set of applied joint moments. Such an understanding simplifies and guides the subsequent control design and experimentation process. The forward dynamics algorithms also provide the necessary analytical foundation for conducting analysis and simulation studies. The multiarm algorithms are based on the filtering and smoothing approach recently advanced by the author [1] for single-arm dynamics, and they can be built up modularly from the single-arm algorithms. The algorithms compute recursively the joint-angle accelerations, the contact forces and the task-object accelerations. Algorithms are also developed to evaluate in closed form the linear transformations from the active joint moments to the joint-angle accelerations, to the task-object accelerations and to the task-object contact forces. A possible computing architecture is presented as a precursor to a more complete investigation of the computational performance of the dynamics algorithms.

CONTENTS

1. INTRODUCTION 1

2. CONFIGURATION AND PROBLEM STATEMENT 3

3. STATES AND CO-STATES 4

4. FORCE, VELOCITY, AND ACCELERATION CONSTRAINTS 5

5. RECURSIVE KINEMATICS 6

6. RECURSIVE DYNAMICS 11

7. TWO-POINT BOUNDARY-VALUE PROBLEM 12

8. FORWARD DYNAMICS 13

9. ALGORITHM ARCHITECTURE 19

10. CLOSED-FORM MAPS FROM JOINT MOMENTS TO CONTACT FORCES AND JOINT-ANGLE ACCELERATIONS 23

11. RELATIONSHIP TO OTHER WORK 29

12. CONCLUDING REMARKS AND FUTURE DIRECTIONS 32

ACKNOWLEDGMENT 32

REFERENCES 33

FIGURES

2.1 Multiple Arms Holding a Common Task Object 3

9.1 Five-State Filtering and Smoothing Architecture 19

9.2 Spoked-Wheel Computing Architecture Concept 20

9.3 Contact Forces as a Weighted Sum of Free Tip Forces and Task-Object Bias Force 21

10.1 Three-Stage Evaluation of Closed-Chain Influence Matrix 29

PRECEDING PAGE BLANK NOT FILMED

1. INTRODUCTION

This report solves the problem of forward dynamics for multiple possibly redundant robot arms operating on a common task object. By doing this, it extends to a multiarm closed-chain system the approach of spatially recursive filtering and smoothing introduced by the author in [1] and used there to solve single-arm dynamics problems. Cooperating arms are useful in tasks that require carrying heavy loads and manipulating cumbersome objects. Such tasks could exceed the force and work envelope limits of single arms. The problem of forward dynamics is to find the task-object mass center acceleration and the contact forces, given the active joint moments. The forward dynamics problem also includes finding the corresponding set of joint-angle accelerations.

The main motivation for solving the forward dynamics problem is to understand the dynamical behavior of the multiple-arm system. Such an understanding makes it easier to develop more insightful control algorithms. This is true, in particular, if the aim is to use simple control schemes. Typically, the simpler the control scheme the better the understanding of the plant model has to be. A fundamental understanding of the forward dynamics problem also makes it easier to conduct experimental studies and to correct possible anomalies.

The spatially recursive filtering and smoothing methods of [1], when extended to multiple arms, lead to a very simple statement and solution of the forward dynamics problem. In this solution, almost every computational step has a corresponding geometrical or physical interpretation. This provides valuable insights into how the underlying physics and geometry of the problem affect the resulting algorithm structure. Another feature of the filtering and smoothing approach is that it organizes the computations required to solve the forward dynamics problem into a highly developed and well-understood framework. This framework (which includes the Riccati equation, Kalman gains, covariances, prediction, correction, etc.) has been highly successful in other application areas. This report does not aim to advance the algorithms for their computational efficiency, since the main contribution of the report is to enhance analytical understanding of the way multiple arms behave dynamically. Results on computational performance of the algorithms will be reported subsequently.

The forward dynamics solution consists of several parallel sequences, one for each arm. Each of the sequences consists of the following five stages:

1. An inward filtering stage begins at the task object and proceeds sequentially from link to link to the base. This first stage is identical to that in the single-arm problem in [1]. It uses the active joint moments to compute a set of filtered state (spatial force) estimates and a residual error process. It also generates a set of Kalman gains.

2. Outward smoothing begins at the base and terminates at the task-object contact points. This second stage uses the residual process emerging from the first stage to compute the free-tip accelerations and the corresponding inertial D'Alembert forces that would exist in the absence of the task object. It also computes the equivalent spatial inertia of the multiple-arm system (minus the task object) as seen from the arm tips.

3. Computation of the contact forces imparted to the task object by the arms is the third stage. Communication between the parallel sequences for each of the arms occurs in this stage. This is the only place in the five-stage computation in which interaction between the simultaneous (one per arm) sequences occurs. As an option, the contact forces can be used to compute the task-object mass center accelerations.

4. Inward filtering computes corrections (due to the contact forces) to the residual process resulting from the first step above.

5. Outward smoothing computes the desired joint-angle accelerations based on the corrected residual process. This final smoothing stage is identical to the smoothing stage in the single-arm solution [1].

This five-stage solution for several arms is an extension of the two-stage solution for single arms in [1]. The first and the fifth stage above coincide with the single-arm solution. These two stages are mutually adjoint [2]. The second, third, and fourth stages involve additional computations to evaluate the contact forces and their effect on the residual process. The second and fourth stages are also mutually adjoint. The single-arm solutions of [1] provide the necessary building blocks to solve the multiple-arm problem. If the closed chain is broken (if the task object is dropped for example) the multiple-arm solution can be modified easily. The intermediate steps $2-4$ above, in which the contact forces and their effects are computed, are not performed. Elimination of these three steps in the multiple-arm solution results in solving the forward dynamics problem for several independent, mechanically decoupled arms.

If the above filtering/smoothing/filtering/smoothing process is performed symbolically, a closed-form equation results, which involves what will be referred to as an influence matrix. This matrix relates the active joint moments to the joint-angle accelerations. Similar matrices relate the active joint moments to the contact forces and to the task-object mass center acceleration. This is in an extension of the results of [1] that produce a closed-form expression for the inverse of the composite multilink system inertia matrix in terms of smoothed state and co-state estimation error covariances. The influence matrices involved in the multiple-arm system can also be evaluated recursively in terms of the estimation error covariances.

The remaining sections of the report describe respectively the multiarm closed-chain system, states

and co-states, constraints, recursive kinematics and dynamics, two-point boundary-value problem, forward dynamics, algorithm architecture, closed-form influence matrices, relationship to other work, and concluding remarks.

2. CONFIGURATION AND PROBLEM STATEMENT

Consider the closed-chain mechanical system illustrated in Fig. 2.1. The system is intended to represent several arms moving a commonly held task object. The number of arms is denoted by NA. Each of the arms has N links numbered $1, \ldots, N$ and N joints also numbered $1, \ldots, N$. The arms are identified with the index i. The last joint of each of the arms is labeled N and is attached to an immobile base. The mass center of the task object is denoted by the symbol C. Each of the arms is attached to the task object at a contact point labeled 0. Perfect attachment is assumed. This implies that there is no relative motion at the contact points and that there is perfect transmission of constraint forces between the arms and the task object. There is no loss of generality due to this assumption because extension is simple to situations in which relative motion between the task-object and the arms is allowed at the contact points. Each of the contact points is at a fixed and known location on the task object.

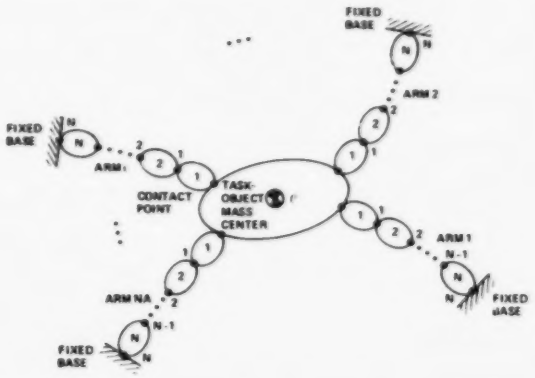

Fig. 2.1 Multiple Arms Holding a Common Task Object

Let link k in the i^{th} arm be characterized by a rotational inertia tensor $I_i(k)$ about joint k, a mass $\rho_i(k)$, a vector $L_i(k)$ from joint k to joint $k-1$, and a vector $c_i(k)$ from joint k to the link k mass center. The 6×6 spatial inertia matrix [1] is defined as

$$m_i(k) = \begin{pmatrix} I_i(k) & \rho_i(k)\tilde{c}_i(k) \\ -\rho_i(k)\tilde{c}_i(k) & \rho_i(k)U \end{pmatrix}$$

This matrix summarizes both the translational and rotational inertia properties of link k about joint k.

The matrix U is the 3×3 unit matrix. The composite $6NA \times 6NA$ matrix

$$M(k) = \begin{pmatrix} m_1(k) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & m_{NA}(k) \end{pmatrix}$$

characterizes the combined inertia properties of all (one per arm) links k in the overall multiarm system.

Let joint k in the i^{th} arm be characterized by a unit vector $h_i(k)$ along its axis of rotation. The joints are assumed to be rotational, although joints allowing relative translation between adjoining links can be handled easily with the same approach. The 6-dimensional unit vectors $h_i(k)$ can be used to obtain the scalar joint moment from the 6-dimensional vector of spatial (rotational plus linear) forces. Define also the composite vector $H(k) = [h_1^T(k), \ldots, h_{NA}^T(k)]$.

Let $\tau_i(k)$ be the active moment applied about the axis of joint k. Define the NA-dimensional vector $T(k) = [\tau_1(k), \ldots, \tau_{NA}(k)]$. Let $\theta_i(k)$ be the corresponding joint angle, which is positive in the right hand sense about $h_i(k)$. Define the NA-dimensional vector $\Theta(k) = [\theta_1(k), \ldots, \theta_{NA}(k)]$. The corresponding joint-angle velocities and accelerations are defined by $\dot{\Theta}(k)$ and $\ddot{\Theta}(k)$.

The objective is to define a recursive method for computation of the joint-angle accelerations $\ddot{\Theta}(k)$, the task-object mass center acceleration, and the contact forces on the task object. This problem is to be solved given the values of $\Theta(k)$, $\dot{\Theta}(k)$, $M(k)$, $L_i(k)$, $H(k)$ and $T(k)$.

3. STATES AND CO-STATES

At any given time instant, the relationship between applied joint moments and the resulting joint-angle accelerations is linear. In the single-arm analysis of [1], a spatially recursive state space model has been introduced to characterize this relationship. The state space approach can also be used for the multiple-arm problem as outlined below.

The state space model involves the definition, at each joint k, of a 6×1 state vector $x(k)$ equal to the spatial force [1] on the negative side, toward the arm tip, of that joint. For the i^{th} arm, this spatial force is denoted by $x_i(k)$. This is a vector of three moments and three linear forces acting on the i^{th} arm link k and due to i^{th} arm link $k+1$. The subscript i implies that the associated state $x_i(k)$ corresponds to the i^{th} arm. Let $x(k) = [x_1(k), \ldots, x_{NA}(k)]$ be a composite $6NA \times 1$ vector formed by the NA spatial forces $x_i(k)$. The active joint moments $T(k) = [\tau_1(k), \ldots, \tau_{NA}(k)]$ and the spatial forces $x(k)$ are related by $T(k) = H(k)x(k)$.

The spatial velocity [1] on the negative side of joint k is denoted by $v_i(k)$. This is a 6×1 vector of three angular velocities and three linear velocities. The index i is again used to denote the i^{th} arm. The composite velocity $V(k) = [v_1(k), \ldots, v_{NA}(k)]$ is a $6NA \times 1$ vector made up of the spatial velocities $v_i(k)$

at joint k of each of the arms.

A spatial acceleration $\lambda(k)$ is also defined at each joint. The spatial acceleration is recognized as a 6×1 co-state vector of the type commonly encountered in optimal control and estimation problems [1]. The spatial accelerations $\lambda_i(k)$ are obtained by proper time differentiation [1] of the spatial velocities $v_i(k)$. The composite acceleration $\lambda(k) = [\lambda_1(k), \ldots, \lambda_{NA}(k)]$ is a $6NA \times 1$ vector made up of the accelerations $\lambda_i(k)$.

One of the key ideas introduced in [1] is the use of the states and co-states defined above to propagate recursively the spatial forces and accelerations. This propagation takes place between distinct spatial locations (between the tip and base, for example). The following 6×6 transition matrix

$$\phi_i(k,j) = \begin{pmatrix} U & \tilde{L}_i(k,j) \\ 0 & U \end{pmatrix}$$

is used to propagate forces (states) inwardly from the task object to the base [1]. Its transpose $\phi_i^T(k,k-1)$ can be used to propagate velocities and accelerations (co-states) in an outward direction. In both of these matrices, $L_i(k,j)$ is the vector from joint k to joint j, and $\tilde{L}_i(k,j)$ denotes the cross-product operation $L_i(k,j) \times (\cdot)$. This matrix satisfies the following properties:

$$\phi_i(k,j) = \phi_i(k,m)\phi_i(m,j); \quad \phi_i(k,j) = \phi_i^{-1}(j,k); \quad \phi_i(k,k) = U$$

which state that the matrix satisfies the semigroup property, that the matrix can be inverted by interchanging its two arguments, and that the matrix becomes the identity if its two arguments coincide.

The composite multiple-arm transition matrix

$$\phi(k,j) = \begin{pmatrix} \phi_1(k,j) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \phi_{NA}(k,j) \end{pmatrix} \tag{3.1}$$

is used to express a simultaneous transition from joint j to joint k in all of the arms. This composite transition matrix will be used to describe simultaneous recursions (one recursion per arm) to propagate forces, velocities, and accelerations for the multiple-arm system.

4. FORCE, VELOCITY, AND ACCELERATION CONSTRAINTS

This section summarizes the constraints that the contact forces, velocities, and accelerations must satisfy because the task object is rigid. In addition, these constraints are combined for use in subsequent sections of the report. The constraints can be summarized as follows:

$$\textit{Force Balance} \quad x(C) = A(C,0)x(0) \tag{4.1}$$

$$\textit{Velocity Balance} \quad V(0) = A^T(C,0)v(C) \tag{4.2}$$

$$\textit{Acceleration Balance} \quad \lambda(0) = A^T(C,0)\alpha(C) \tag{4.3}$$

$$\textit{Task-Object Spatial Motion} \quad x(C) = M(C)\alpha(C) + b(C) \tag{4.4}$$

in which the $6 \times 6NA$ matrix $A(C,0)$ is defined as

$$A(C,0) = [\phi_1(C,0), \ldots, \phi_{NA}(C,0)] \tag{4.5}$$

Equation (4.1) states that the net spatial force $x(C)$ acting at the task-object mass center equals the weighted sum of the contact forces $x_i(0)$ at the contact points. The contact forces must be propagated from the contact points to the task-object mass center by means of the matrices $\phi_i(C,0)$ in the transformation $A(C,0)$.

Equation (4.2) relates the composite spatial velocity $V(0) = [v_1(0), \ldots, v_{NA}(0)]$ at the contact points and the spatial velocity $v(C)$ at the task-object mass center. This equation follows from the observation that $v_i(0) = \phi_i^T(C,0)v(C)$. These observations are true because the task object is a rigid body. Equation (4.3), relating the composite acceleration $\lambda(0)$ at the contact points and the spatial acceleration $\alpha(C)$ at the task-object mass center, follows [1] from differentiating Eq. (4.2) with respect to time.

Equation (4.4) is the equation of spatial (translational and rotational) motion of the task object about its mass center. It is expressed in terms of the spatial inertia $M(C)$ of the task object and the spatial bias force $b(C)$ acting on this object.

Equations (4.1), (4.3), and (4.4) can be combined into

$$\lambda(0) = A^T(C,0)M^{-1}(C)[A(C,0)x(0) - b(C)] \tag{4.6}$$

This equation relates the spatial force $x(0)$ and the spatial acceleration $\lambda(0)$ at the contact points. It will be used in Sec. 8 to obtain a solution to the multiple-arm forward dynamics problem.

5. RECURSIVE KINEMATICS

This section provides the kinematic relationships necessary to determine velocities and accelerations for the entire system, given only partial (joint-angle velocities and accelerations, for instance) velocity and acceleration information. This will allow the focus of the forward dynamics problem solved in Sec. 8 to be that of determining joint-angle accelerations from the applied joint moments. Other accelerations, such as task-object mass center accelerations, can then be determined by combining the results of Sec. 8 and of this section. This section solves the following four closely related problems:

1. Determine the spatial velocity at each of the joints of the arms, given the joint-angle velocities (RESULTS 5.1 and 5.2).

2. Determine the velocity of the task-object mass center, given the joint-angle velocities (RESULT 5.3).

3. Determine the joint-angle velocities, given the task-object mass center velocity (RESULTS 5.4 and 5.5).

4. Determine a set of constrained joint-angle velocities, given a set of possibly unconstrained joint-angle velocities (RESULT 5.6).

Similar relationships (RESULT 5.7) are also outlined among the spatial accelerations, the joint-angle accelerations, and the task-object mass center acceleration.

RESULT 5.1. *The sequence of spatial velocities* $V(k) = [v_1(k), \ldots, v_{NA}(k)]$ *can be obtained from the joint-angle velocities* $\dot{\Theta}(k) = [\dot{\theta}_1(k), \ldots, \dot{\theta}_{NA}(k)]$ *by means of the following outward recursion*

LOOP $k = N, \ldots, 1$**;**

$$V(k) = \phi^T(k+1,k)V(k+1) + H^T(k)\dot{\Theta}(k) \tag{5.1}$$

END LOOP;

with the terminal velocity $V(N+1) = 0$. *This is an outward sequence that starts at the base of the arms and goes to joint* 1 *of each of the arms. There are* NA *parallel sequences generated by the above equations, one for each arm. The composite notation* $V(k) = [v_1(k), \ldots, v_{NA}(k)]$ *involving* NA *vectors is used to describe* NA *parallel sequences with a single set of recursive equations. The spatial velocity* $V(0) = \phi^T(1,0)V(1)$ *at the contact points can be computed after the end of the sequence.*

The above is a simple extension of the results of [1] on single-arm recursive kinematics. Equation (5.1) can be cast in more compact notation. To this end, define the lower triangular matrix

$$\Phi = \begin{pmatrix} I & 0 & \cdots & 0 \\ \phi(2,1) & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \phi(N,1) & \phi(N,2) & \cdots & I \end{pmatrix}$$

Similarly, define the matrices $H = diag[H(1), \ldots, H(N)]$ and $B^T = [\phi^T(1,0), 0, \ldots, 0]$.

RESULT 5.2. *The spatial velocities* $V = [V(1), \ldots, V(N)]$ *are*

$$V = \Phi^T H^T \dot{\Theta} \tag{5.2}$$

where $\dot{\Theta} = [\dot{\Theta}(1), \ldots, \dot{\Theta}(N)]$. *At the contact points, the spatial velocity is*

$$V(0) = B^T \Phi^T H^T \dot{\Theta} \tag{5.3}$$

Proof: Equation (5.1) implies that $V(k) = \sum_{i=k}^{N} \phi^T(i,k) H^T(i) \dot{\Theta}(i)$. This proves Eq. (5.2). Observe that $V(0) = \phi^T(1,0) V(1)$ to prove Eq. (5.3).

The contact point velocities $V(0)$ that emerge from the above are not necessarily compatible with the condition that the task object be a rigid body. To enforce this condition, the velocity balance constraint of Eq. (4.2) must be imposed. This constraint states that the velocity $V(0)$ of the contact points and the velocity $v(C)$ of the task-object mass center are related by $V(0) = A^T(C,0) v(C)$. Use of this in Eq. (5.3) leads to

$$B^T \Phi^T H^T \dot{\Theta} = A^T(C,0) v(C) \tag{5.4}$$

This constraint is a linear system of $6NA$ scalar equations in the $6NA$ unknown joint-angle velocities. The constraint can be used to determine the task-object mass center velocity from the joint-angle velocities.

RESULT 5.3. *The velocity $v(C)$ of the task-object mass center can be determined from*

$$v(C) = [A(C,0) A^T(C,0)]^{-1} A(C,0) B^T \Phi^T H^T \dot{\Theta}$$

The prescribed joint-angle velocities must satisfy the constraint of Eq. (5.4).

Proof: Premultiply Eq. (5.4) by A and solve for $v(C)$.

Now consider the inverse problem of obtaining a set of joint-angle velocities $\dot{\Theta}$, given the spatial velocity $v(C)$ of the task object. There are two approaches to this problem. The first approach assumes that none of the arms is redundant. It also assumes that none of the arms is at a Jacobian singularity. The second approach leads to a minimum-norm solution, in which the joint-angle velocity vector $\dot{\Theta}_n$ that satisfies the constraint of Eq. (5.4) and that has smallest norm is determined. These two solutions are outlined in Results 5.4 and 5.5 below.

RESULT 5.4. *The joint-angle velocity $\dot{\Theta}$ and the task-object mass center velocity $v(C)$ are related by*

$$\dot{\Theta} = (B^T \Phi^T H^T)^{-1} A^T(C,0) v(C)$$

This solution follows from Eq. (5.4) if the matrix $B^T \Phi^T H^T$ is invertible. For redundant arms, there may be more than 6 joint angles per arm. The linear system of scalar equations summarized by Eq. (5.4) therefore may have more unknowns than equations. The solution to Eq. (5.4) may not be unique. An alternative and unique solution (that also applies to redundant arms) is provided below. This solution is one which minimizes the norm of the joint-angle velocity vector $\dot{\Theta}$ subject to the constraint of Eq. (5.4). If the matrix $B^T \Phi^T H^T$ is invertible, then the minimum-norm solution reduces to the simpler solution of Result 5.4.

RESULT 5.5. *The joint-angle velocities $\dot{\Theta}_o$ of smallest norm satisfying the constraint of Eq. (5.4) are*

$$\dot{\Theta}_o = H\Phi BQ^{-1}(0)A^T(0,C)v(C) \tag{5.5}$$

in which $Q(0)$ is defined as

$$Q(0) = B^T\Phi^T H^T H\Phi B = \sum_{i=1}^{N} \phi^T(i,0)H^T(i)H(i)\phi(i,0) \tag{5.6}$$

The corresponding spatial velocity vector is $V = \Phi^T H^T \dot{\Theta}_o$. The minimum-norm solution can also be generated recursively by means of the outward/inward sequence

LOOP $k = N, \ldots, 1$;

$$Q(k) = \phi^T(k+1,k)Q(k+1)\phi(k+1,k) + H^T(k)H(k) \tag{5.7}$$

END LOOP;

with the terminal condition $Q(N+1) = 0$. The velocity covariance $Q(0) = \phi^T(1,0)Q(1)\phi(1,0)$ at the contact points is computed at the end of this outward sequence. This is used to initialize the inward sequence

$$X(0) = Q^{-1}(0)A^T(C,0)v(C)$$

LOOP $k = 1, \ldots, N$;

$$\begin{aligned} X(k) &= \phi(k,k-1)X(k-1) \\ \dot{\Theta}_o(k) &= H(k)X(k) \\ \dot{\Theta}_p(k) &= \dot{\Theta}(k) - \dot{\Theta}_o(k) \end{aligned} \tag{5.8}$$

END LOOP;

The vector $\dot{\Theta}_o$, of minimum norm, as well as the error $\dot{\Theta}_p$ (the difference between unconstrained and constrained joint-angle velocities), result from this outward sequence.

Proof: Use standard techniques for finding minimum-norm solutions to show Eq. (5.5). Observe that $Q(0)$ defined in Eq. (5.6) can be computed by means of Eq. (5.7). To show Eq. (5.8), define the composite state as $X = [X(1), \ldots, X(N)]$. Observe that $X = \Phi BQ^{-1}A^T v(C)$ and use the transition matrix properties, in Eq. (3.1), of the matrix ϕ in Φ.

The problem is now addressed of determining a set of constrained joint-angle velocities $\dot{\Theta}_o$, given a set of possibly unconstrained joint-angle velocities $\dot{\Theta}$. This problem arises, for example, if joint-angle velocities have been computed independently for each of the arms, without taking into account the mechanical coupling due to the task object. The solution to this problem is summarized in the following result, which is obtained by combining Results 5.1, 5.3, and 5.5 above.

RESULT 5.6. *A set of joint-angle velocities $\dot{\Theta}_o$, which satisfy the constraint of Eq. (5.4) that the task object be rigid, can be obtained from a set of possibly unconstrained joint-angle velocities $\dot{\Theta}$ by means of the relationship*

$$\dot{\Theta}_o = H\Phi B(B^T Q B)^{-1} A^T (AA^T)^{-1} AB^T \Phi^T H^T \dot{\Theta}$$

The constrained joint-angle velocities $\dot{\Theta}_o(k)$ can be found recursively from the unconstrained joint-angle velocities $\dot{\Theta}(k)$ by means of the following outward/inward sequence:

1. Use Eq. (5.1) to determine the contact point velocities $V(0)$. Use Eq. (5.7) to determine the contact point velocity covariance. Do both of these with an outward recursion.

2. Use Result 5.3 to determine the task-object mass center velocity.

3. Use Eq. (5.8) to determine the joint-angle accelerations of smallest norm.

Spatial Accelerations

The above analysis applies to the spatial velocities. A very similar analysis applies to the sequence of spatial accelerations. Because of this similarity, only the results are presented without including the detailed analysis that leads to the results.

The composite spatial acceleration vector $\lambda = [\lambda(1), \ldots, \lambda(N)]$ at all of the joints can be generated by means of

$$\lambda = \Phi^T (H^T \ddot{\Theta} + \eta)$$

At the contact points 0, the spatial acceleration is

$$\lambda(0) = B^T \Phi^T (H^T \ddot{\Theta} + \eta)$$

The acceleration vector can also be generated recursively by

LOOP $k = N, \ldots, 1$;

$$\lambda(k) = \phi^T(k+1, k)\lambda(k+1) + H^T(k)\ddot{\Theta}(k) + \eta(k)$$

END LOOP;

with the terminal condition $\lambda(N+1) = 0$. The acceleration $\lambda(0)$ at the contact points can be computed from $\lambda(0) = \phi^T(1,0)\lambda(1)$. The symbol $\eta(k)$ denotes the spatial acceleration bias [1] due to coriolis and other nonlinear velocity-dependent effects. This bias term is assumed to have been computed prior to solving the forward dynamics problem.

The joint-angle accelerations $\ddot{\Theta}$ satisfy the constraints

$$B^T\Phi^T(H^T\ddot{\Theta} + \eta) = A^T(C,0)\lambda(C)$$

If the matrix $B^T\Phi^T H^T$ is invertible, then

$$\ddot{\Theta} = (B^T\Phi^T H^T)^{-1}[A^T(C,0)\lambda(C) - B^T\Phi^T\eta]$$

If $B^T\Phi^T H^T$ is not invertible, then use the minimum-norm solution

$$\ddot{\Theta}_m = H\Phi B(B^TQB)^{-1}[A^T(C,0)\lambda(C) - B^T\Phi^T\eta]$$

The solution of minimum norm $\ddot{\Theta}_m$ can be obtained from a possibly unconstrained set of joint-angle accelerations $\ddot{\Theta}$ by means of

$$\ddot{\Theta}_m = H\Phi B(B^TQB)^{-1}A^T(AA^T)^{-1}AB^T\Phi^T H^T\ddot{\Theta}$$

The minimum-norm solution for joint-angle accelerations can also be generated recursively by a set of equations similar to those in Result 5.5.

6. RECURSIVE DYNAMICS

The sequence of spatial forces $x(k)$ satisfies the recursive equations:

LOOP $k = 1, \ldots, N$;

$$x(k) = \phi(k, k-1)x(k-1) + M(k)\lambda(k) + b(k)$$

$$T(k) = H(k)x(k)$$

END LOOP;

with the initial state $x(0)$ at the contact points. This initial state corresponds to the contact forces imparted to the task object by the arms. The contact forces are assumed to be initially unknown. They are determined as part of the solution to the forward dynamics problem, as explained in Sec. 8.

The above recursive dynamics equations are derived in detail in [1]. Only an outline of their derivation is presented here. The state propagation equation follows [1] by writing the equations of rotational and translational motion for a set of typical links k, one for each arm, and by applying Newton's third law at joints k. The output equation represents the state-to-output map that projects the $6NA$-dimensional state vector $x(k)$ into the set of scalars $T(k)$. The equations compute recursively the spatial forces $x(k)$ for the set of parallel inward sequences, one along each arm. The sequences begin at the contact points and terminate at the base. The sequences produce as output the active joint moments $T(k)$ along the joint axes. The spatial accelerations $\lambda(k)$ are viewed as a known input to the sequence. The spatial bias forces $b(k)$ defined in [1] are also assumed to be known.

The relationship between the initial state, the spatial accelerations, and the output joint moments can be expressed in the more compact notation

$$T = H\Phi Bx(0) + H\Phi(M\lambda + b)$$

in which $T = [T(1), \ldots, T(N)]$ and $\lambda = [\lambda(1), \ldots, \lambda(N)]$ are respectively the vector of active joint moments and the vector of spatial accelerations.

7. TWO-POINT BOUNDARY-VALUE PROBLEM

The sequences of forces $x(k)$ and accelerations $\lambda(k)$ satisfy the following two-point boundary-value problem:

$$\lambda(0) = A^T M^{-1}(C)[Ax(0) - b(C)] \tag{7.1}$$

LOOP $k = 1, \ldots, N$;

$$x(k) = \phi(k, k-1)x(k-1) + M(k)\lambda(k) + b(k)$$

$$T(k) = H(k)x(k)$$

END LOOP;

$$\lambda(N) = H^T(N)\ddot{\Theta}(N) \tag{7.2}$$

LOOP $k = N, \ldots, 1$;

$$\lambda(k-1) = \phi^T(k, k-1)\lambda(k) + H^T(k-1)\ddot{\Theta}(k-1) + \eta(k-1)$$

END LOOP;

This is a two-point boundary-value problem in the sense that the boundary conditions of Eqs. (7.1) and (7.2) are satisfied at two sets of points: the contact points 0 where the task object is attached to the arms, and the support points where the arms are mounted to the immobile base. The boundary condition of Eq. (7.1) can be said to be mixed, because it requires that the states and co-states be constrained by a linear relationship at the task-object contact points. The relationship between states and co-states has been derived in Eq. (4.6) from the spatial equation of motion for the task object about its mass center and from the force and acceleration constraints due to rigidity of the task object. When combined with Eq. (7.2), Eq. (7.1) defines a set of mixed-fixed boundary conditions. These boundary conditions differ from the free-fixed conditions of the single-arm problem studied in [1]. There, the boundary conditions correspond to a situation in which the state vanishes at the arm tips and the co-state vanishes at the base. Consequently, the mixed-fixed boundary-value problem above has some features that are not present in the free-fixed case of [1]. Nonetheless, two-point boundary-value problems in which the states and co-states are related at the boundary have been investigated in [6]. The same general methods of [6] are applied in the following section to find a recursive solution to the two-point boundary-value problem.

8. FORWARD DYNAMICS

The aim here is to solve the two-point boundary-value problem by means of the recursive methods of filtering and smoothing. These algorithms are the extension to closed-chain multiple arms of the techniques advanced in [1] for single arms.

RESULT 8.1. *The joint-angle accelerations $\ddot{\Theta}$, the task-object contact forces $x(0)$, and the task-object acceleration $\alpha(C)$ can be computed by means of the following five-stage sequence:*

1. *Filtering of Active Joint Moments to Compute Free Innovations*

The first stage, which is identical to that of [1], computes a set of free filtered state estimates, a set of free innovations, and a set of Kalman gains. It also computes the scalar inertia of the composite body outboard of each joint. This inertia is referred to as the joint inertia [1]. The free innovations, the joint inertias, and the Kalman gains are stored for use in subsequent stages.

$$\textit{Initial Conditions} \quad z_{FREE}(0) = 0; \quad P(0) = 0; \quad G(1,0) = 0$$

LOOP $k = 1, \ldots, N$;

$$\textit{Gain} \quad G(k,k-1) = \phi(k,k-1)P(k-1)H^T(k-1)D^{-1}(k-1)$$

$$\textit{Filter Transition Matrix} \quad \psi(k,k-1) = \phi(k,k-1) - G(k,k-1)H(k-1)$$

$$\textit{Filtered State} \quad z_{FREE}(k) = \psi(k,k-1)z_{FREE}(k-1) + G(k,k-1)\tau(k-1) + b(k) \tag{8.1}$$

$$\textit{Spatial Inertia} \quad P(k) = \psi(k,k-1)P(k-1)\psi^T(k,k-1) + M(k)$$

$$\textit{Joint Inertia} \quad D(k) = H(k)P(k)H^T(k)$$

$$\textit{Free Innovations} \quad e_{FREE}(k) = D^{-1/2}(k)[\tau(k) - H(k)z_{FREE}(k)]$$

END LOOP;

2. *Smoothing of Free Innovations to Compute Free Tip Accelerations and D'Alembert Forces*

This stage takes the sequence of free innovations emerging from the filter and produces the free joint-angle accelerations $\ddot{\Theta}_{FREE}(k)$ *that would, in the absence of the task object, result from application of the active joint moments. It also computes the corresponding spatial accelerations* $\lambda_{FREE}(k)$.

$$\textit{Terminal Conditions} \quad \lambda_{FREE}(N+1) = 0; \quad \Lambda(N+1) = 0$$

LOOP $k = N, \ldots, 1$;

$$\textit{Joint-Angle Acceleration} \quad \ddot{\Theta}_{FREE}(k) = D^{-1/2}(k)e_{FREE}(k) - G^T(k+1,k)\lambda_{FREE}(k+1)$$

$$\textit{Co-State} \quad \lambda_{FREE}(k) = \psi^T(k+1,k)\lambda_{FREE}(k+1) \tag{8.2}$$

$$\textit{Co-State Covariance} \quad \Lambda(k) = \psi^T(k+1,k)\Lambda(k+1)\psi(k+1,k) + H^T(k)D^{-1}(k)H(k)$$

END LOOP;

The free tip accelerations $\lambda_{FREE}(0)$ *and the free co-state covariance* $\Lambda(0)$ *emerge at the end of this stage. These can be used to determine the D'Alembert forces*

$$x_{FREE}(0) = \Lambda^{-1}(0)\lambda_{FREE}(0)$$

due to the free tip accelerations. For simplicity, the bias acceleration $\eta(k)$ *has been set to zero in arriving at Eq. (8.2). The algorithms in [1] show how to account for this acceleration.*

3. *Contact Forces and Task-Object Accelerations From Free Tip Accelerations and Task-Object Bias Force*

The contact forces $x(0)$ *are*

$$\textit{Contact Forces} \quad x(0) = \Omega^{-1}[\lambda_{FREE}(0) + A^T M^{-1}(C)b(C)] \tag{8.3}$$

in which $\Omega = A^T M^{-1}(C)A + \Lambda(0)$. *The outputs of this stage are the contact forces acting on the task object. It is assumed that the matrix* $A^T M^{-1}(C)A$ *involved in the matrix* Ω *has been computed in advance and is available from storage. The corresponding task-object accelerations are obtained from*

$$\alpha(C) = M^{-1}(C)[Ax(0) - b(C)]$$

If only the task-object contact forces and accelerations are desired, stop here. The remaining two stages are needed only if the joint-angle accelerations are required.

4. *Filtering of Contact Forces to Correct Innovations*

This stage determines the incremental changes in the spatial force estimates at all of the joints due to the forces $x(0)$ *at the task-object contact points. It also modifies the residual process to account for these contact forces.*

$$\textit{Initial State Increment} \quad \delta z(0) = x(0)$$

LOOP $k = 1, \ldots, N$;

$$\textit{State Increment} \quad \delta z(k) = \psi(k, k-1)\delta z(k-1)$$

$$\textit{Modified Innovations} \quad e(k) = e_{FREE}(k) - D^{-1/2}(k)H(k)\delta z(k) \tag{8.4}$$

END LOOP;

5. *Smoothing of Modified Innovations to Compute Joint-Angle Accelerations*

This last stage is identical to the smoothing stage for the single-arm case of [1]. It consists of an outward sequence that processes the modified residual process in order to obtain the closed-chain joint-angle accelerations.

$$\textit{Terminal Co-State} \quad \lambda(N+1) = 0$$

LOOP $k = N, \ldots, 1$;

$$\textit{Joint-Angle Accelerations} \quad \ddot{\Theta}(k) = D^{-1/2}(k)e(k) - G^T(k+1,k)\lambda(k+1)$$

$$\textit{Co-State} \quad \lambda(k) = \psi^T(k+1,k)\lambda(k+1) + H^T(k)\ddot{\Theta}(k) \tag{8.5}$$

END LOOP;

Proof: The proof is based on the sweep method of [6], suitably modified for problems in which the states and co-states are constrained by a linear relationship at one of the boundaries. This method begins with the assumption that the states and co-states are related by

$$x(k) = z_{FREE}(k) + \delta z(k) + P(k)[\lambda_{FREE}(k) + \delta\lambda(k)] \tag{8.6}$$

The filtered state estimate is composed of two solutions: (1) a nominal solution $z_{FREE}(k)$ that satisfies the filtering equations with "free" initial conditions $z_{FREE}(0) = 0$. This solution is the sequence of filtered state estimates that would be obtained if the task object were not present and the arms were therefore not mechanically coupled; (2) a correction $\delta z(k)$ to the filtered state estimate due to the nonzero task-object contact forces $x(0)$. These contact forces do not become known until completion of the third stage. Similarly, the co-state solution is partitioned into a nominal component $\lambda_{FREE}(k)$ and an increment $\delta\lambda(k)$ due to the nonzero contact forces $x(0)$. The nominal co-states $\lambda_{FREE}(k)$ correspond to the problem that would arise if the arms were mechanically decoupled, whereas the incremental correction $\delta\lambda(k)$ is due to the presence of the task object.

Substitution of Eq. (8.6) in the two-point boundary-value problem of Sec. 7 implies that: $z_{FREE}(k)$ and $P(k)$ satisfy Eq. (8.1); the filtered state increments $\delta z(k)$ satisfy Eq. (8.4); the nominal co-states satisfy Eq. (8.2); and the total co-states satisfy Eq. (8.5). The total co-states are defined as the sum of the nominal co-states and the co-state increments.

The unknown initial condition $x(0)$ is determined by first observing that

$$z_{FREE}(0) + \delta z(0) + P(0)[\lambda_{FREE}(0) + \delta\lambda(0)] = x(0)$$

To satisfy this boundary condition, let $z_{FREE}(0) = 0$, $P(0) = 0$ and $\delta z(0) = x(0)$. Then, use Eq. (8.4) to obtain that the increments $\delta e(k)$ to the innovations $e_{FREE}(k)$ are given by

$$\delta e(k) = -D^{-1/2}(k)H(k)\psi(k,0)x(0)$$

in which $\psi(k,m)$ is defined as

$$\psi(k,m) = \prod_{i=m+1}^{k} \psi(i,i-1)$$

This matrix can be shown [1] to be the transition matrix for the Kalman filter going from the negative side of joint m to the negative side of joint k. Joint k is assumed to be inboard, toward the base, of joint m in this definition.

Now, use Eq. (8.5) to obtain that the co-state increment at the contact points is given by

$$\delta\lambda(0) = -\Lambda(0)x(0)$$

This together with the constraint of Eq. (4.6) implies that the contact forces are given by Eq. (8.3). This will be shown in more detail in Result 10.3. Equation (8.3) determines the contact forces $x(0)$ in terms of the "free" contact accelerations $\lambda_{FREE}(0)$ and the bias force $b(C)$ at the task-object mass center. The equation, immediately following Eq. (8.3), that determines the task-object accelerations follows by combining the constraints of Eqs. (4.1) and (4.4).

In Eq. (8.4), these contact forces are used to modify the free residual process e_{FREE} that has been computed in Eq. (8.2). The modified residual process e that emerges from Eq. (8.4) is used in Eq. (8.5), the smoothing equations of the last stage, to compute the closed-chain joint-angle accelerations.

Linear Operator Notation

To gain further insight on the forward dynamics algorithm, the preceding result can be recast in terms of linear operator notation [2]. This is done by "integrating" the discrete difference equations in Eqs. (8.1)–(8.5) using the transition matrix $\psi(k,m)$ of the Kalman filter. To this end, define the composite matrices

$$\Psi = \begin{pmatrix} I & 0 & \cdots & 0 \\ \psi(2,1) & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \psi(N,1) & \psi(N,2) & \cdots & I \end{pmatrix}; \quad S = \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ I & 0 & \cdots & 0 & 0 \\ 0 & I & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & I & 0 \end{pmatrix}$$

The matrix Ψ can be viewed [2] as a linear operator that characterizes the composite response (at all of the joints) of the Kalman filter. The matrix S is a shift operator, which will be used typically to account for one-step spatial "delays" that occur in Eq. (8.1)–(8.5), the discrete difference equations defining the filter and smoother. Define also the block diagonal matrices $G = diag[G(1,0),\ldots,G(N,N-1)]$, $H = diag[H(1),\ldots,H(N)]$, and $D = diag[D(1),\ldots,D(N)]$.

RESULT 8.2. *The closed-chain multiarm joint-angle accelerations $\ddot{\Theta}$ can be determined from the applied joint moments T by means of the following linear operator equation*

$$\ddot{\Theta} = (I - L^T)D^{-1/2}(I - K\Omega^{-1}K^T)D^{-1/2}(I - L)T \tag{8.7}$$

in which L and K are lower triangular matrices defined as

$$L = H\Psi GS$$

$$K = D^{-1/2}H\Psi B$$

and L^T amd K^T are their corresponding transpose matrices. For simplicity, the bias forces b and bias accelerations η are assumed to be zero.

Proof: Integrate the relationships in Eq. (8.1) to obtain $z_{FREE}(k) = \sum_{i=1}^{k-1} \psi(k,i+1)G(i+1,i)T(i)$. This implies that

$$e_{FREE} = D^{-1/2}(I - L)T \tag{8.8}$$

in which e_{FREE} is the free innovations process. Similarly, the smoothing equations of Eq. (8.2) imply that

$$\lambda_{FREE}(0) = K^T e_{FREE}$$

in which $K^T = B^T \Psi^T H^T D^{-1/2}$. This determines the contact point accelerations that would be there if the task object did not exist and the arms were therefore mechanically decoupled. The corresponding contact point forces are determined by the equation

$$x(0) = \Omega^{-1} \lambda_{FREE}(0)$$

in which $\Omega = \Lambda(0) + A^T M^{-1}(C) A$. These contact forces are used as an initial condition in the fourth stage to compute the resulting forces at all of the joints.

Note that Eq. (8.4) implies $\delta z = \Psi B x(0)$ in which δz is the vector of filtered state increments due to the nonzero contact forces. This has a corresponding residual process increment of $\delta e = -D^{-1/2} H \Psi B x(0)$. Hence,

$$\delta e = -K x(0)$$

The modified innovations process, defined as $e = e_{FREE} + \delta e$, is therefore given by

$$e = (I - K\Omega^{-1} K^T) e_{FREE} \tag{8.9}$$

Integrate Eq. (8.5), the last smoothing stage, to obtain

$$\ddot{\Theta} = (I - L^T) D^{-1/2} e \tag{8.10}$$

Put together Eqs. (8.8)–(8.10) to obtain the desired result.

Satisfaction of Acceleration Constraints

The objective here is to verify that the constraints imposed on the closed-chain joint-angle accelerations by the presence of the task-object are satisfied by the accelerations computed in Results 8.1 and 8.2.

RESULT 8.3. *The closed-chain joint-angle accelerations computed by Results 8.1 and 8.2 satisfy the constraint*

$$B^T \Phi^T H^T \ddot{\Theta} = A^T \alpha(C)$$

in which $\alpha(C)$ is the acceleration of the task-object mass center.

Proof: Multiply Eq. (8.7) by $B^T \Phi^T H^T$ to obtain

$$\lambda(0) = B^T \Psi^T H^T D^{-1/2} (I - K\Omega^{-1} K^T) e$$

In arriving at this, the identity $(I - L) H \Phi B = H \Psi B$ established in [2] has been used. However, the definition of K above implies that $\lambda(0) = (I - K^T K \Omega^{-1}) K^T e_{FREE}$. Since $K^T K = \Lambda(0)$ and $\Omega =$

$\Lambda(0) + A^T M^{-1}(C)A$, then $\lambda(0) = A^T M^{-1}(0)Ax(0)$ in which the condition $x(0) = \Omega^{-1} K^T e_{FREE}$ has been used. However, recall that the net component of force $x(C)$ at the task-object mass center is defined by $x(C) = Ax(0)$. In addition, $M^{-1}(C)x(C) = \alpha(C)$. These last three equations together imply the desired result.

9. ALGORITHM ARCHITECTURE

The architecture of the forward dynamics algorithms summarized by Results 8.1 and 8.2 is illustrated in Fig. 9.1. The algorithms can be subdivided into five major stages.

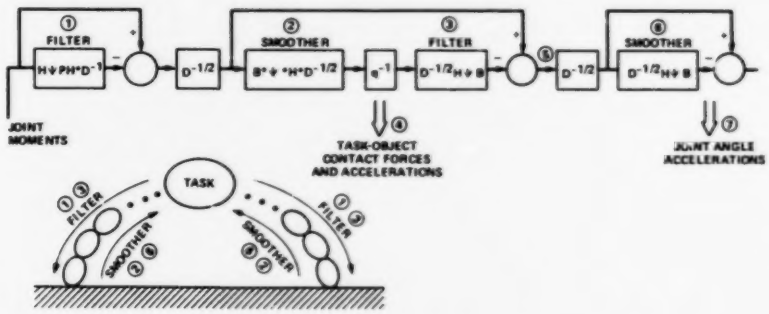

Fig. 9.1 Five-Stage Filtering and Smoothing Architecture

The first is a filtering stage (1) that begins at the tips of the arms. The task object is assumed not to exist. The filter computes a sequence of spatial force estimates that would exist for a free multiple-arm system unconstrained by the presence of the task object. There are several such sequences, one for each arm.

The next three stages compute the modified residual process by means of the equation $e = (I - K\Omega^{-1}K^T)e_{FREE}$. The second (2) and fourth (3) stages are mutually adjoint, since they are defined respectively by the operator K and its corresponding transpose K^T. The second stage involves a smoothing operation K^T on the residuals to obtain the contact point forces (4). It also computes the corresponding free accelerations at the contact points that would exist if the arms were mechanically decoupled from each other. The contact forces are computed in the third stage. The contact forces are used as an initial condition by the fourth stage to determine a sequence of modified residuals (5).

The fifth stage (6) takes the modified residuals as an input and computes the sequence of joint-angle accelerations (7). This smoothing stage begins at the base of the arms and terminates at the contact points. The first and fifth stages are mutually adjoint.

A possible "spoked-wheel" computing architecture for the filtering and smoothing algorithms is illustrated in Fig. 9.2.

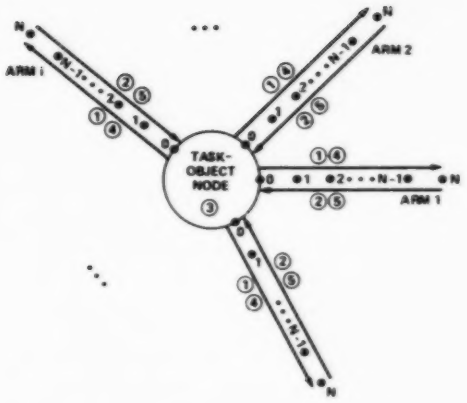

Fig. 9.2 Spoked-Wheel Computing Architecture Concept

The first stage describes parallel sequences (1), one per arm, that go from the task object to the base of the arms. The sequences are independent of each other. They can be implemented by means of separate computational processes that do not communicate with each other. The sequence in each arm is the same as would be used for single-arm forward dynamics. Therefore, if a computer program already exists to solve the forward dynamics problem for one arm, this program can be reproduced to solve the problem for multiple arms. The second stage also involves simultaneous parallel sequences (2). Each of the sequences goes from the base of an arm to the corresponding contact point where the same arm is attached to the task object. As in the first stage, they are independent of each other and can be implemented in parallel processes that do not communicate while the spatial recursions are being conducted.

Interprocess communication (3) among arms occurs in order to compute the task-object contact forces and accelerations. This is the only place in the five-stage sequence that communication takes place.

The fourth stage is very similar, but not identical, to the first stage. The main difference between the two is that the fourth stage does not require spatial inertia computations, whereas the first one does. Inspection of Eqs. (8.1) and (8.3) shows this difference. The fourth stage uses the contact forces to start simultaneous sequences (4) that go from the task-object contact points to the arm base. The fifth and final stage (5) is very similar to the second stage. The main difference between the two is that the fifth stage does not require evaluation of the co-state covariance matrix. This is made evident by comparison of Eqs. (8.2) and (8.5). Because there is no communication among the arms in the fourth and fifth stages, separate computational processes can be used.

Predictor/Corrector Architecture to Determine Contact Forces

The aim here is to examine more thoroughly the third stage of the forward dynamics algorithm of Sec. 8. In this stage, the free tip accelerations $\lambda_{FREE}(0)$ and the corresponding co-state covariance $\Lambda(0)$, that would exist at the tips of the arms in the absence of the task object, are used to compute the contact

forces. This computation occurs in Eq. (8.3). This equation is used here as a starting point to explore in more detail the relationship between free accelerations and the contact forces.

To this end, recall that the free D'Alembert forces $x_{FREE}(0)$ are defined as

$$x_{FREE}(0) = P\lambda_{FREE}(0)$$

in which $P = \Lambda^{-1}(0)$. These are the inertial forces that exist at the tips of the arms, due to the free tip accelerations λ_{FREE}. The matrix P reflects the composite spatial inertia of the arms as seen from the arm tips. With these definitions at hand, Eq. (8.3) is recast in a form that shows that the contact forces are a weighted linear combination of the free tip forces $x_{FREE}(0)$ and the bias force $b(C)$ at the task-object mass center. This is done in the following result, which is obtained by rearranging Eq. (8.3).

RESULT 9.1. *The contact forces $x(0)$ imparted by the arms on the task object can be expressed as*

$$x(0) = x_{FREE}(0) + \mathcal{G}[b(C) - Ax_{FREE}(0)] \tag{9.1}$$

in which $\mathcal{G}$ is the Kalman gain

$$\mathcal{G} = PA^T[APA^T + M(C)]^{-1}$$

An alternative expression for the Kalman gain is

$$\mathcal{G} = [\Lambda(0) + A^T M^{-1}(C)A]^{-1} A^T M^{-1}(C)$$

Equation (9.1) has the predictor/corrector architecture of the Kalman filter [1]. This is illustrated in Fig. 9.3.

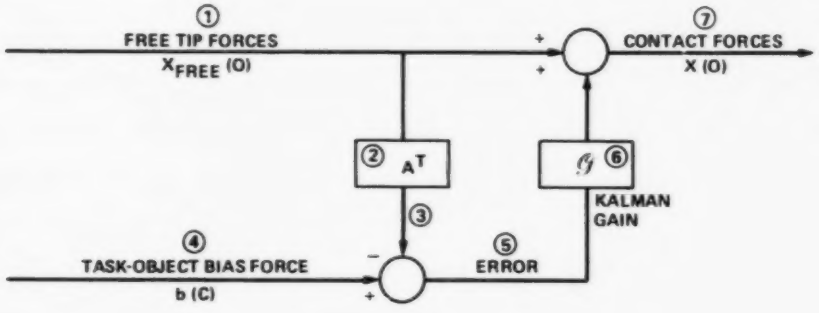

Fig. 9.3 Contact Forces as a Weighted Sum of Free Tip Forces and Task-Object Bias Force

In this architecture, the contact forces $x(0)$ are a weighted sum of the free tip forces and the bias force at the task-object mass center. The Kalman gain can be viewed as an optimal weighting matrix that

combines the arm spatial inertia P (as seen from the arm tips) and the task-object inertia $M(C)$ about its mass center. The free tip force $x_{FREE}(0)$ (1) is a predicted estimate. This predicted estimate has an inherent error with a covariance P. The predicted estimate is multiplied by the matrix A (2) to obtain the effect of the free tip force at the task-object mass center. The multiplication by A can therefore be interpreted as a step in which the state estimate $x_{FREE}(0)$ is projected into a measurement space defined at the task-object mass center. At this stage, there are two estimates for the spatial force that occurs at the mass center of the task object: (1) an estimate $Ax_{FREE}(0)$ (3) due to the acceleration and inertia of the arms and (2) an estimate $b(C)$ (4) due to the spatial force bias resulting from gyroscopic and other task-object velocity-dependent effects. Both of these estimates have a built-in uncertainty made quantitative by their respective estimation error covariances P and $M(C)$. The second estimate can be viewed as a measurement that is used to update the first estimate. From these two estimates, an error term (5) is created that, after being multiplied by the Kalman gain (6), is used to correct the predicted estimate. The contact forces (7) result from this correction step.

Further physical insight is gained by examining the behavior of the Kalman gain and of the resulting estimates in two opposite extreme cases: (1) the spatial inertia $M(C)$ of the task-object is large compared to the arm spatial inertia P seen from the tips of the arms and (2) the inertia P is much larger than the inertia $M(C)$. In the first case, the Kalman gain vanishes. Therefore, the correction term also vanishes. The contact forces $x(0)$ are then equal to the free tip forces $x_{FREE}(0)$. In this situation, the task object does not move, and the contact forces are equal to those that correspond to the tips of the arms being rigidly attached to an immobile object. In the second case, in which the matrix P is much larger than the matrix $M(C)$, the projection $A\mathcal{G}$ of the Kalman gain along the state-to-measurement transformation A approaches the unit matrix. The corrected estimate $Ax(0)$ in the same direction depends only on the bias spatial force $b(C)$, and the effect of the free tip forces vanishes. This follows by multiplying Eq. (9.1) by the state-to-measurement matrix A.

Predictor/Corrector Architecture to Determine Task Object Accelerations

A similar predictor/corrector approach can be used to compute the task-object accelerations. To this end, define

$$\alpha_{BIAS} = -M^{-1}(C)b(C)$$

These are the accelerations that the task-object mass center would undergo, if the task object were disconnected from the arms. The bias acceleration is due only to the bias force $b(C)$. The task-object mass center acceleration $\alpha(C)$ is given by a weighted linear combination of the bias acceleration α_{FREE} and the free accelerations $\lambda_{FREE}(0)$ that would exist at the tips of the arms, if the task object were not being held

and the arms were therefore mechanically decoupled.

RESULT 9.2. *The task-object mass center accelerations are*

$$\alpha(\mathcal{C}) = \alpha_{FREE} + \mathcal{G}^T[\lambda_{FREE}(0) - A^T\alpha_{FREE}] \tag{9.2}$$

in which $\mathcal{G}$ is the Kalman gain defined in Eq. (9.1).

Proof: Combine Eqs. (4.1), (4.4), and (9.1) and solve for $\alpha(\mathcal{C})$.

This result states that the acceleration at the task-object mass center can be computed by using a predictor/corrector approach. The bias acceleration α_{FREE} plays the role of the predicted estimate, whereas the free tip acceleration $\lambda_{FREE}(0)$ at the arm tips plays the role of the measurement. An error term $\lambda_{FREE}(0) - A^T\alpha_{FREE}$ is used to correct the predicted estimate. This is the central feature of Eq. (9.2). The relative weighting given this error term is determined by the Kalman gain $\mathcal{G}$. The corrected estimate, corresponding to the actual task-object mass center acceleration, is the sum of the bias acceleration and the appropriately weighted error term.

Concluding Remarks About Architecture

In the algorithm architecture of Figs. 9.1, 9.2, and 9.3, every computational step has a corresponding physical interpretation. This contributes toward elegance, efficiency, and reliability of the forward dynamics algorithms. For example, the Riccati equation propagates inertia. Another example is that the idea of prediction followed by correction is pervasive throughout the algorithm. It occurs in the fundamental single-arm solution that makes up the first two stages of the algorithm. It occurs again in going from the first two stages to the last three stages. The first two stages can be viewed as a prediction step in a more global computation in which the free joint-angle accelerations are first predicted assuming that the task object does not exist. The remaining three stages correct the joint-angle accelerations for the contact forces that, in fact, do exist and which have been disregarded in the first two stages.

10. CLOSED-FORM MAPS FROM JOINT MOMENTS TO CONTACT FORCES AND JOINT-ANGLE ACCELERATIONS

The aim here is to obtain in closed form the linear transformations from the active joint moments to the task-object contact forces and from the active joint moments to the joint-angle accelerations. These results are the extension to closed-chain multiple arms of the results obtained in [1] for inversion of the composite inertia matrix of open-chain single arms. In order to set the background for developing this extension, two of the results of [1] are first reviewed. This is done in Results 10.1 and 10.2, which follow.

RESULT 10.1. *The free joint-angle accelerations $\ddot{\Theta}_{FREE}(k)$ produced, in the absence of the task object, by the active joint moments $T(i)$ is*

$$\ddot{\Theta}_{FREE}(k) = \sum_{i=1}^{N} W_{FREE}(k,i)T(i)$$

in which the off-diagonal elements of the kernel W_{FREE} are

	$i < k$	$i > k$
$W_{FREE}(k,i)$	$\tilde{H}(k)\psi(k,i+1)G(i+1,i)$	$G^T(k+1,k)\psi^T(i,k+1)\tilde{H}^T(i)$

with the modified state-to-output map $\tilde{H}(k)$ defined as

$$\tilde{H}(k) = G^T(k+1,k)\Lambda(k+1)\psi(k+1,k) - D^{-1}(k)H(k)$$

The diagonal elements are $W_{FREE}(k,k) = D^{-1}(k) + G^T(k+1,k)\Lambda(k+1)G(k+1,k)$.

Proof: Only an outline of the proof is presented. Details are contained in [1]. Recall that the free joint-angle accelerations $\ddot{\Theta}_{FREE}$ and the active joint moments T are related by

$$\ddot{\Theta}_{FREE}(k) = D^{-1/2}(k)e_{FREE}(k) - G^T(k+1,k)\sum_{i=k+1}^{N}\psi^T(i,k+1)H^T(i)D^{-1/2}(i)e_{FREE}(i) \qquad (10.1)$$

$$e_{FREE}(i) = D^{-1/2}(i)[T(i) - \sum_{j=1}^{i-1}H(i)\psi(i,j+1)G(j+1,j)T(j)] \qquad (10.2)$$

Substitution of Eq. (10.2) into Eq. (10.1) leads to the desired result.

The kernel $W_{FREE}(k,i)$ above is the general element of the matrix W_{FREE} that relates the free joint-angle accelerations and the active joint moments by $\ddot{\Theta}_{FREE} = W_{FREE}T_{FREE}$. The result states that the joint-angle acceleration at a given joint k is a weighted sum of the active joint moments at all of the joints. The weighting kernel $W_{FREE}(k,i)$ represents the influence that the moment $T(i)$ at joint i has on the acceleration $\ddot{\Theta}_{FREE}(k)$ at joint k. Because of this, the corresponding matrix W_{FREE} will be referred to as the free influence matrix. It has been shown in [1] that this matrix is the inverse of the composite multilink system inertia matrix. The next result [1] shows how to compute this free influence matrix recursively.

RESULT 10.2. *The influence matrix $W_{FREE}(k,i)$ for the multiarm open-chain system can be computed by means of the following inward/outward sequence:*

Diagonal Sequence of Kalman Gains

Solution of the discrete Riccati equation along the diagonal $i = k$ of the square region $1 \leq i,k \leq N$ produces a set of Kalman gains and joint inertias.

$$P(0) = 0; \quad G(1,0) = 0$$

LOOP $i = 1, \ldots, N$;

$$G(i,i-1) = \phi(i,i-1)P(i-1)H^T(i-1)D^{-1}(i-1)$$

$$P(i) = \psi(i,i-1)P(i-1)\psi^T(i,i-1) + M(i) \tag{10.3}$$

$$D(i) = H(i)P(i)H^T(i)$$

END LOOP;

The gains $G(i,i-1)$ *and the joint inertias* $D(i)$ *are stored in this stage.*

Free Influence Matrix via Sequence of Vertical Sweeps

The free influence matrix is evaluated in the triangular region $1 \leq k \leq i \leq N$ *via successive vertical sweeps of diminishing length. A vertical sweep is defined as a sequence generated by varying* k *from the diagonal in which* $k = i$ *to the bottom edge of the triangular region in which* $k = 1$*. The index* i *is held constant at a fixed value for each vertical sweep. Successive vertical sweeps are generated by varying* i *repeatedly from* $i = N$ *to* $i = 1$*.*

$$\Lambda(N+1) = 0$$

LOOP $i = N, \ldots, 1$;

$$W_{FREE}(i,i) = D^{-1}(i) + G^T(i+1,i)\Lambda(i+1)G(i+1,i)$$

$$\tilde{H}(i) = G^T(i+1,i)\Lambda(i+1)\psi(i+1,i) - D^{-1}(i)H(i)$$

$$\Lambda(i) = \psi^T(i+1,i)\Lambda(i+1)\psi(i+1,i) + H(i)D^{-1}(i)H^T(i)$$

$$\lambda(i) = \tilde{H}^T(i) \tag{10.4}$$

LOOP $k = i-1, \ldots, 1$;

$$W_{FREE}(k,i) = G^T(k+1,k)\lambda(k+1)$$

$$\lambda(k) = \psi^T(k+1,k)\lambda(k+1)$$

END k **LOOP;**

END i **LOOP;**

The free influence matrix elements $W_{FREE}(k,i)$ in the triangular region $1 \le k \le i \le N$ and the co-state covariance $\Lambda(1)$ at joint 1 of each arm are the end result of this stage. The co-state covariance $\Lambda(0)$ at the contact points can be computed by means of $\Lambda(0) = \phi^T(1,0)\Lambda(1)\phi(1,0)$.

The free joint-angle accelerations $\ddot{\Theta}_{FREE}(k)$ produce the accelerations $\lambda_{FREE}(k)$, which in the absence of the task object, would exist at the various joints k of the arms. These accelerations are of particular interest because they can be used to evaluate the tip contact forces that the arms impart to the task object.

RESULT 10.3. *The contact forces $x(0)$ and the free tip accelerations $\lambda_{FREE}(0)$ are related by*

$$x(0) = \Omega^{-1}\lambda_{FREE}(0) = -\Omega^{-1}\sum_{j=1}^{N}\psi^T(j,0)\tilde{H}(j)T(j) \tag{10.5}$$

in which Ω is defined by

$$\Omega = A^T M^{-1}(0)A + \Lambda(0)$$

Proof: Use Eq. (8.1), the state equations in the filtering stage, to obtain

$$e_{FREE}(i) = D^{-1/2}(i)[T(i) - H(i)\sum_{j=1}^{i-1}\psi(i,j+1)G(j+1,j)T(j)] \tag{10.6}$$

Similarly, use Eq. (8.2), the co-state equations in the smoothing stage, to obtain

$$\lambda_{FREE}(0) = \sum_{i=1}^{N}\psi^T(i,0)H^T(i)D^{-1/2}(i)e_{FREE}(i) \tag{10.7}$$

Substitute Eq. (10.6) into Eq. (10.7), use the identity $\sum_{i=1}^{N}\sum_{j=1}^{i-1} = \sum_{j=1}^{N-1}\sum_{i=j+1}^{N}$ and recall that $\Lambda(i)$ is defined as $\Lambda(i) = \sum_{j=i}^{N}\psi^T(j,i)H^T(j)D^{-1}(j)H(j)\psi(j,i)$ to obtain the desired result.

The next result determines the joint-angle accelerations $\ddot{\Theta}(k)$ for the closed-chain system in terms of the free joint-angle accelerations $\ddot{\Theta}_{FREE}(k)$ and the contact forces $x(0)$.

RESULT 10.4. *The closed-chain multiarm joint-angle accelerations $\ddot{\Theta}(k)$, the free joint-angle accelerations $\ddot{\Theta}_{FREE}(k)$, and the contact forces $x(0)$ are related by*

$$\ddot{\Theta}(k) = \ddot{\Theta}_{FREE}(k) + \tilde{H}(k)\psi(k,0)x(0) \tag{10.8}$$

Proof: Use Eq. (8.5) to show that

$$\ddot{\Theta}(k) = D^{-1/2}(k)e(k) - G^T(k+1,k)\sum_{i=k+1}^{N}\psi^T(i,k+1)H^T(i)D^{-1/2}(i)e(i) \tag{10.9}$$

in which $e(i)$ is the modified residual process given by

$$e(i) = e_{FREE}(i) - D^{-1/2}(i)H(i)\psi(i,0)x(0) \tag{10.10}$$

Substitute Eq. (10.10) into Eq. (10.9) and use Eq. (10.2).

RESULT 10.5. *The closed-chain joint-angle accelerations* $\ddot{\Theta}(k)$ *are related to the active joint moments* $T(k)$ *by means of*

$$\ddot{\Theta}(k) = \ddot{\Theta}_{FREE}(k) - \sum_{i=1}^{N} \Delta W(k,i) T(i) \tag{10.11}$$

in which

	$i \leq k$	$i \geq k$
$\Delta W(k,i)$	$\tilde{H}(k)\psi(k,i)S(i)\tilde{H}^T(i)$	$\tilde{H}(k)S(k)\psi^T(i,k)\tilde{H}^T(i)$

The matrix $S(i)$ *is defined as* $S(i) = \psi(i,0)\Omega^{-1}\psi^T(i,0)$. *This matrix satisfies the inward recursion*

LOOP $i = 1, \ldots, N$;

$$S(i) = \psi(i,i-1)S(i-1)\psi^T(i,i-1)$$

END LOOP;

with the initial condition $S(0) = \Omega^{-1}$.

Proof: Substitute Eq. (10.5) into Eq. (10.8).

RESULT 10.6. *The closed-chain joint-angle accelerations* $\ddot{\Theta}(k)$ *and the active joint moments* $T(i)$ *are related by*

$$\ddot{\Theta}(k) = \sum_{i=1}^{N} W(k,i) T(i)$$

in which

$$W(k,i) = W_{FREE}(k,i) - \Delta W(k,i)$$

Proof: Combine Results 10.1 and 10.5.

This result states that the influence matrix for the closed-chain system can be obtained by subtracting a correction term from the free influence matrix of the open-chain system. The following result states how to compute this correction by means of a sequence of vertical sweeps.

RESULT 10.7. *The closed-chain influence matrix* $W(k,i)$ *can be computed by means of the following succession of vertical sweeps covering the upper triangular region* $1 \leq i \leq k \leq N$:

$$S(0) = [A^T M^{-1}(0) A + \Lambda(0)]^{-1}$$

LOOP $i = 1, \ldots, N$;

$$S(i) = \psi(i, i-1)S(i-1)\psi^T(i, i-1)$$

$$x(i) = S(i)\tilde{H}^T(i) \tag{10.12}$$

LOOP $k = i, \ldots, N$;

$$W(k,i) = W_{FREE}(k,i) - \tilde{H}(k)x(k)$$

$$x(k+1) = \psi(k+1, k)x(k)$$

END k **LOOP**;

END i **LOOP**;

A vertical sweep is a sequence generated by varying k from $k = i$ to $k = N$, while keeping i constant. Successive vertical sweeps are produced by varying the index i along the diagonal from $i = 1$ to $i = N$. The free influence matrix elements $W_{FREE}(k,i)$ are assumed to have been evaluated, using Result 10.2 for example, prior to start of the inward sequence.

Proof: This follows from the definition of $\Delta W(k,i)$ in Eq. (10.11).

The closed-chain influence matrix can be evaluated by a three-stage inward/outward/inward sequence obtained by the combination of Results 10.2 and 10.7. These three stages are illustrated in Fig. 10.1.

The first stage is governed by Eq. (10.3) and is illustrated in Fig. 10.1(a). It evaluates the Kalman gains along the diagonal of the square region defined by $1 \leq k, i \leq N$. The first stage begins at the task-object contact points and goes to the base of the arms. There are several parallel sequences generated simultaneously, one for each arm. The gains are stored for use in the second stage.

The second stage, shown in Fig. 10.1(b), computes the elements of the free influence matrix in the lower triangular region $1 \leq i \leq k \leq N$. This stage is described by Eq. (10.4). It can be decomposed into two closely related sequences: (1) a sequence that computes the co-state covariance along the diagonal $i = k$. This sequence also computes the modified state-to-output map $\tilde{H}(i)$ and the diagonal elements of the free influence matrix; (2) a sequence of co-states to compute the off-diagonal elements $W_{FREE}(k,i)$ of the free influence matrix in the triangular region $1 \leq k \leq i \leq N$. This triangular region is covered by a series of successive vertical sweeps. Each vertical sweep starts at the diagonal and proceeds toward the bottom edge of the triangular region. Each vertical sweep is one step shorter than the immediately preceding sweep. The second stage also computes the co-state covariance $\Lambda(0)$ at the contact points between the arms and the task object. This matrix is combined with a term $A^T M^{-1}(0)A$, related to the inverse of the task-object inertia, in order to form the matrix $\Omega = A^T M^{-1}(0)A + \Lambda(0)$. This matrix is used to initialize the third stage.

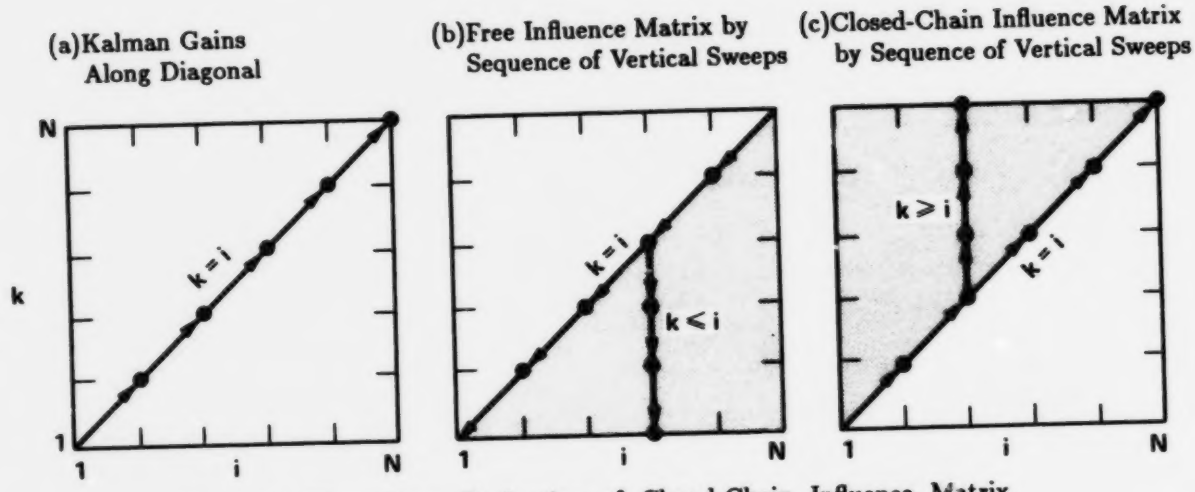

Fig. 10.1 Three-Stage Evaluation of Closed-Chain Influence Matrix

The third stage, described by Eq. (10.12), is an inward sequence from the task object to the base of the arms. This sequence is illustrated in Fig. 10.1(c). It is initialized with the initial condition $S(0) = \Omega^{-1}$ at the contact points. The stage has two related subsequences: (1) a diagonal sequence that evaluates the matrices $S(i)$ along the diagonal and (2) a series of vertical sweeps that evaluate the off-diagonal elements $W(k,i)$ in the upper triangular region $1 \leq i \leq k \leq N$. Each vertical sweep is one step shorter than the previous vertical sweep. Symmetry of the closed-chain influence matrix is used to evaluate the matrix in the lower triangular region $1 \leq k \leq i \leq N$. This makes the results of the second and third stages compatible.

11. RELATIONSHIP TO OTHER WORK

The main contribution of this report is to extend to closed-chain multiple arms the results of [1–3] on the application of spatially recursive filtering techniques to robot dynamics. The report also provides a more thorough analysis of the dual-arm problem, which is a special case of the results presented here, than an earlier dual-arm dynamics paper by the author [4].

Robot arm dynamics problems are easily solved using filtering and smoothing techniques[1–4]. The techniques are effective and simple to use because they are based on a methodology [5–8] that has resulted from many years [7] of research on filtering and smoothing for linear state space systems. This foundation is exploited here to solve the multiple-arm forward dynamics problem. Two main results are obtained: recursive algorithms to compute joint-angle accelerations, task-object accelerations, and task-object contact forces from applied joint moments; and closed-form evaluation of the influence matrices that relate the applied moments and the resulting closed-chain accelerations and contact forces. As discussed in Sec.

9, the recursive multiarm forward dynamics algorithms can be built up modularly from the single-arm solutions of [1].

Multibody system dynamics have been studied by a variety of other methods [9–27]. Typical applications (in addition to robotics) are spacecraft and ground vehicles. For example, [9–11] use graph theory combined with Newton-Euler mechanics to analyze multibody systems of arbitrary topology. Similarly, [12] combines the methods from linear network analysis and mechanics to study very general collections of bodies. Much of the work on multibody dynamics has aimed at developing computer programs [13–15] for automated assembly of equations of motion. Perhaps the most comprehensive of these programs is in [15], which aims to analyze arbitrary systems of rigid and flexible bodies. In general, with a few exceptions, these methods are not recursive. The emphasis is on first evaluating a composite multibody system inertia matrix and then on inverting this matrix.

Because of their focus on spatial recursions, [16,17] are in the same spirit as the present report. Reference [16] introduced a spatial algebra, related to screw theory, to analyze single-arm robot forward dynamics. The work of [17] extended this to closed-chain configurations. The single-arm algorithms of [1,16] are similar, but not identical. The main difference is that [1] recognizes the equivalence of the forward dynamics algorithms with the methods of filtering and smoothing. This equivalence allows the extension from open to closed chains to be performed within the context of solving two-point boundary-value problems in which the states and co-states are related at one of the boundaries. Such mixed boundary-value problems and their corresponding solutions are very well understood [6]. This provides a very mature foundation to address the multiple-arm dynamics problem and to interpret the resulting algorithms. Such a foundation does not appear to be readily available for the approach of [16,17]. There, the recursive algorithms and their properties have to be rediscovered and redeveloped from general principles (mathematical induction, for instance) without the benefit of insights and relationships that emerge naturally from the filtering and smoothing methods [5–8]. For example, casting the equation that accumulates spatial inertia as a Riccati equation, a step taken in [1] but not in [16], makes it possible to use much of what is known, (i.e., numerical stability properties and approaches to retain symmetry of its solution) about this equation. Similarly, knowing that there is a quantity in the forward dynamics algorithms that is analogous to the innovations process of linear filtering theory [7,8] allows use of well-known properties of this process (its whiteness, for instance) to check for computational errors. Another example of the relationships provided by the filtering and smoothing approach is the predictor/corrector computation (see Sec. 9) of the contact forces as a weighted linear combination of the free tip D'Alembert forces and the task-object mass center bias force.

Significant results on closed-chain dynamics have emerged also from research in multilegged walking

machines and multiple manipulators [18−23]. The dynamics of walking machines are very similar to those of multiple arms. The work in [18-23] appears to be the most advanced investigation to date of the issues of modeling, simulation, and control of a closed-chain mechanical system. It is more advanced in the sense that extensive investigations [20−22] have been conducted on issues in control as well as optimal force distribution [23]. While control issues are beyond the scope of the present report, they will be addressed in the future.

The dynamical analysis [18,19] underpinning the work in [20] is somewhat different from the analysis presented here. The main difference is that in [18,19] the forward dynamics problem is addressed somewhat indirectly by application of methods (such as recursive Newton-Euler) that have been developed primarily for inverse dynamics. In [18,19] the Newton-Euler approach of [24] is applied to each chain of a multiple-chain system to determine the coefficient matrices for a set of linear equations. These matrices are then combined to obtain a completely determined set of linear equations for the entire system. This approach is recursive in the sense that the coefficient matrices are evaluated by means of spatial recursions. Here, the forward dynamics problem is addressed more directly. The spatial recursions for each arm solve directly for the accelerations in terms of the applied joint moments, without the need to evaluate a set of coefficient matrices. The spatial recursions (involving Kalman filtering, smoothing, Riccati equations, etc.) for each arm are tailored to address the forward dynamics problem directly without going through the intermediate step of solving the inverse dynamics problem.

However, although the filtering and smoothing techniques solve the forward dynamics problem for each arm recursively and directly, batch-mode inversion of a matrix (or, equivalently, solution of a set of linear equations) cannot be completely avoided. This inversion must take place in order to compute the contact forces at the task object, given the free tip accelerations that would result if the task object did not exist. This step occurs in Eq. (9.1). The set of linear algebraic equations that must be solved in this step is related, but not identical, to the set of equations that must be solved in [18,19]. The main difference is that Eq. (9.1) is only of dimension $6(NA+1)$, whereas the dimension of the set of equations in [18,19] may be higher. For the special case of two arms in a closed chain, the computations involved in determining contact forces can be reduced even further by taking spatial recursions that go from the left arm base to the right arm base and then return, in contrast to the recursions used here that start and terminate at the task object [4]. However, the fact that the computations to determine contact forces are simpler than in [18,19] does not necessarily imply that the overall approach used here is necessarily faster than that of [18,19]. The reason is that the spatial recursions of [18,19] to solve the inverse dynamics problem for each chain are simpler than the filtering and smoothing recursions of this report, which require the additional burden of determining spatial inertias recursively. A comparison of computational performance between

the two approaches would be of interest but is outside the scope of this report.

Solution of the forward dynamics problem is but the initial step toward addressing the issues of simulation and control. An important issue in analysis of closed-chain systems is that of reducing the number of degrees of freedom by combining the kinematic constraint equations with the equations of motion. This is investigated in [25–27], which also discuss the problems of numerical integration of the equations of motion. These problems have not been investigated as thoroughly for the forward dynamics methods of this report as they have for current methods. This report, however, provides the necessary analytical foundation for such an investigation.

12. CONCLUDING REMARKS AND FUTURE DIRECTIONS

This report solves the problem of multiple-arm forward dynamics by means of filtering and smoothing algorithms. The algorithms can be used to compute joint-angle and task-object accelerations, given a set of applied joint moments. They can also be used to compute in closed form the linear transformations from joint moments to accelerations. The algorithms are easy to understand and use because they are based on very well understood methods. Multiple-arm algorithms can be built up modularly from single-arm algorithms. This makes it relatively simple to implement the algorithms by reproducing existing algorithms and software. Additional steps are required only to implement the computation of contact forces, since these forces typically do not appear in single-arm forward dynamics problems.

It is not the intent to advance the filtering and smoothing algorithms as being numerically superior to existing methods for solving closed-chain forward dynamics problems. The main benefit of using the algorithms is in the insight and physical understanding that they provide by organizing the forward dynamics computations into a well-established framework.

This report provides the analytical foundation for future work in the following areas:

1. Computational experimentation to establish numerical properties of the forward dynamics algorithms.
2. Control algorithm development based on either the recursive algorithms of Sec. 8 or on the closed-form influence matrices of Sec. 10.
3. Trajectory design and load-balancing algorithms.

ACKNOWLEDGMENT

Thanks are due K. Kreutz for valuable discussions and suggestions for improvement of this report.

REFERENCES

[1] Rodriguez, G., *Kalman Filtering, Smoothing and Recursive Robot Arm Forward and Inverse Dynamics*, JPL Publication 86-48, Pasadena, CA, Dec. 1986.

[2] Rodriguez, G., "Spatially Random Models, Estimation Theory and Robot Arm Dynamics," *Proceedings of Workshop on Space Telerobotics*, JPL Publication 87-13, Pasadena, CA, July 1987.

[3] Rodriguez, G., "Recursive Dynamics of Topological Trees of Rigid Bodies via Kalman Filtering and Bryson-Frazier Smoothing," *VPI/SU Symposium on Dynamics and Control of Large Structures*, Blacksburg, VA, June 1987.

[4] Rodriguez, G., "Recursive Forward Dynamics for Two Robot Arms in a Closed Chain Based on Kalman Filtering and Bryson-Frazier Smoothing," *Robotics*, M. Jamshidi, Ed., North-Holland, 1986.

[5] Kalman, R. E., "A New Approach to Linear Filtering and Prediction Problems," *ASME Transactions, Journal of Basic Engineering*, Vol. D, pp. 35–45, March 1960.

[6] Bryson, A. E., Jr., and Y. C. Ho, *Applied Optimal Control*, Blaisdell Publishing Co., 1969.

[7] Kailath, T., "A View of Three Decades of Linear Filtering Theory," *IEEE Trans. Inf. Theory*, Vol. IT-20, pp.147–181, 1974.

[8] Kailath, T., "The Innovations Approach to Detection and Estimation Theory," *Proc. IEEE*, Vol. 58, pp. 680–695, 1970.

[9] Roberson, R. E., "Adaptation of a General Multibody Dynamical Formalism to Dynamic Simulation of Terrestrial Vehicles," *Vehicle System Dynamics*, Vol. 6, 1977.

[10] Wittenburg, J., *Dynamics of Systems of Rigid Bodies*, Stuttgart: B. G. Teubner, 1977.

[11] Wittenburg, J., "Analytical Methods in Mechanical System Dynamics," *Computer-Aided Analysis and Optimization of Mechanical System Dynamics*, E. J. Huag, Ed., Springer-Verlag, 1984.

[12] Richard, M. J., et al., "Generalized Vector-Network Formulation for the Dynamic Simulation of Multibody Systems," *Journal of Dynamic Systems, Measurement and Control*, Vol. 108, Dec. 1986.

[13] Rosenthal, D. E. and M. A. Sherman, "High Performance Multibody Simulations via Symbolic Equation Manipulation and Kane's Method," *J. Astro. Sciences*, Vol. 34, No. 3, pp. 223–239, July-Sept. 1986.

[14] Frisch, H. P., *A Vector-Dyadic Development of the Equations of Motion for N Coupled Rigid Bodies and Point Masses*, NASA Technical Note D-7767, 1974.

[15] Bodley, C., et al., *A Digital Computer Program for the Dynamic Interaction Simulation of Controls and Structure (DISCOS)*, NASA Technical Paper 1219, Vols. 1 and 2, May 1978.

[16] Featherstone, R., "The Calculation of Robot Dynamics Using Articulated Body Inertias," *The Intl. J. of Robotics Research*, Vol. 2, No. 1, Spring 1983.

[17] Lathrop, R. H., "Constrained (Closed-Loop) Robot Simulation by Local Constraint Propagation," *IEEE International Conference on Robotics and Automation*, San Francisco, CA, 1986.

[18] Orin, D. E. and R. B. McGhee, "Dynamic Computer Simulation of Robotic Mechanisms," *Theory and Practice of Robots and Manipulators*, Warsaw, Poland, Sept. 1981.

[19] Oh, S. Y. and D. Orin, "Dynamic Computer Simulation of Multiple Closed-Chain Robotic Mechanisms," *IEEE International Conference on Robotics and Automation*, San Francisco, CA, 1986.

[20] Orin, D. E. and S. Y. Oh, "Control of Force Distribution in Robotic Mechanisms Containing Closed Kinematic Chains," *Journal of Dynamic Systems, Measurement and Control*, Vol. 103, No. 2, June 1981.

[21] Hemami, H. and F. C. Weimer, "Modeling of Nonholonomic Dynamic Systems With Applications," *Journal of Applied Mechanics*, Vol. 48, March 1981.

[22] Hemami, H. and B. F. Wyman, "Modeling and Control of Constrained Dynamic Systems with Application to Biped Locomotion in the Frontal Plane," *IEEE Transactions on Automatic Control*, Vol. AC-24, No. 4, August 1979.

[23] Nakamura, Y., et al., "Mechanics of Coordinative Manipulation by Multiple Robotic Mechanisms," *IEEE International Conference on Robotics and Automation*, Raleigh, NC, 1987.

[24] Walker, M. W. and D. E. Orin, "Efficient Dynamic Computer Simulation of Robotic Mechanisms," *Journal of Dynamic Systems, Measurement and Control*, Vol. 104, pp. 205–211, Sept. 1982.

[25] Kamman, J. W. and R. L. Huston, "Dynamics of Constrained Multibody Systems," *Journal of Applied Mechanics*, Vol. 51, Dec. 1984.

[26] Haug, E. J., "Elements and Methods of Computational Dynamics," *Computer-Aided Analysis and Optimization of Mechanical System Dynamics*, E. J. Haug, Ed., Springer-Verlag, 1984.

[27] Luh, J. Y. S. and Y. F. Zheng, "Computation of Input Generalized Forces for Robots with Closed Kinematic Chain Mechanisms," *IEEE Journal of Robotics and Automation*, Vol. RA-1, No. 2, June 1985.

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No. JPL PUB 88-6	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Recursive Forward Dynamics for Multiple Robot Arms Moving a Common Task Object		5. Report Date February 15, 1988 6. Performing Organization Code
7. Author(s)		8. Performing Organization Report No.
9. Performing Organization Name and Address JET PROPULSION LABORATORY California Institute of Technology 4800 Oak Grove Drive Pasadena, California 91109		10. Work Unit No. 11. Contract or Grant No. NAS7-918 13. Type of Report and Period Covered JPL Publication
12. Sponsoring Agency Name and Address NATIONAL AERONAUTICS AND SPACE ADMINISTRATION Washington, D.C. 20546		14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

Recursive forward dynamics algorithms are developed for an arbitrary number of robot arms moving a commonly held object. The multiarm forward dynamics problem is to find the angular accelerations at the joints and the contact forces that the arms impart to the task object. The problem also involves finding the acceleration of this object. The multiarm forward dynamics solutions provide a thorough physical and mathematical understanding of the way several arms behave in response to a set of applied joint moments. Such an understanding simplifies and guides the subsequent control design and experimentation process. The forward dynamics algorithms also provide the necessary analytical foundation for conducting analysis amd simulation studies. The multiarm algorithms are based on the filtering and smoothing approach recently advanced by the author (1) for single-armdynamics, and they can be built up modularly from the single-arm algorithms. The algorithms compute recursively the joint-angle accelerations, the contact forces and the task-object accelerations. Algorithms are also developed to evaluate in closed form the linear transformations from the active joint moments to the joint-angle accelerations, to the task-object accelerations and to the task-object contact forces. A possible computing architecture is presented as a precursor to a more complete investigation of the computational performance of the dynamics algorithms.

17. Key Words (Selected by Author(s))	18. Distribution Statement
Robotics	Unclassified; unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified		

JPL 0184 R 9/83